→ Decoder

The decoder is responsible for generating the output sequence one token at a time, using the encoder's output and the previously generated tokens.

Transformer's decoder has 3 main components

① ~~Multi~~ Masked Multi Head Attention

② Multi-head Attention

③ Feed Forward Neural Network

→ Masked Multi Head Attention

Dataset

Eng	French
$<x_1, x_2, x_3>$	$<y_1, y_2>$

$$\Downarrow$$

$<y_1, y_2, 0>$ ⇒ output shifted right

↓ zero padding

Input	output
[4 5 6 7]	[1 2 3]

1- Input Embedding and Positional Encoding

Input $[4 \quad 5 \quad 6 \quad 7]$

output $[1 \quad 2 \quad 3 \quad 0]$

$\downarrow$

4 dimensions

output Embeddings

$[[0.1, 0.2, 0.3, 0.4], [0.5, 0.6, 0.7, 0.8],$
$[0.9, 1.0, 1.1, 1.2], [0, 0, 0, 0]]$

considering positional encoding to be 0

2- Linear Projection for Q, K, V

Query (Q), Key (K), Value (V)

$\cancel{Q = K = V}$ $W^Q = W^K = W^V = I$

Q = Output Embedding $\times W^Q$ = Output Embedding

K = Output Embedding $\times W^K$ = Output Embedding

V = Output Embedding $\times W^V$ = Output Embedding

$Q = K = V = [[0.1, 0.2, 0.3, 0.4], [0.5, 0.6, 0.7, 0.8],$
$[0.9, 1.0, 1.1, 1.2], [0, 0, 0, 0]]$

3- Scaled Dot Product Attention

$$\text{Score} = \frac{Q \times K^T}{\sqrt{d_K}}$$

$$= \frac{Q \times K^T}{2}$$

$$= \frac{\begin{bmatrix} 0.1 & 0.2 & 0.3 & 0.4 \\ 0.5 & 0.6 & 0.7 & 0.8 \\ 0.9 & 1.0 & 1.1 & 1.2 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0.1 & 0.5 & 0.9 & 0 \\ 0.2 & 0.6 & 1.0 & 0 \\ 0.3 & 0.7 & 1.1 & 0 \\ 0.4 & 0.8 & 1.2 & 0 \end{bmatrix}}{2}$$

$$= \begin{bmatrix} 0.3 & 0.7 & 1.1 & 0.0 \\ 0.7 & 1.9 & 3.1 & 0.0 \\ 1.1 & 3.1 & 5.1 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 \end{bmatrix}$$

4- Masked Application

It helps managing the structure of the sequences being processed and ensures the model behaves correctly during training and inference

> Reasons

① Handling variable length sequences with padding MASK.

Purpose

① To handle sequences of different length in batch

② To ensure the padding tokens, which are added to make sequences of uniform length, do not effect the model prediction.

Example

Input Sequence [1, 2, 3]

Output Sequence [4, 5, 0] 0 is padding token

→ Influence attention mechanism

⇓

lead to incorrect or biased prediction

Masking
- → padding mask
- → look ahead mask

Padding Mask → The tokens are ignored

Look Ahead Mask → Maintain auto regressive properties

- → to ensure that each position in the decoder output sequence can only attend the previous position, no future position
- → LanSequence → Language Modeling, Translation

Example Padding Mask

$[4, 5, 0] \xrightarrow[\text{mask}]{\text{padding}} [1, 1, 0]$ 1D

$\begin{bmatrix} 1 & 1 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$ 2D

Token 1 can attend 1,2

Token 2 can attend 1,2

For each token in the sequence, the mask should indicate with token it can attend to

Example Look Ahead Mask

$[4, 5, 0] \longrightarrow [[1\ 0\ 0], [1\ 1\ 0], [1\ 1\ 1]]$

Combine Padding and Look Ahead Mask

Combine Mask = Padding Mask . Look Ahead Mask

$$= \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

wherever in the combined mask, the value is 0, there we specifically specify add -infinity, to zero out the influence when softmax is applied.

> Solving Attention Score with Masking

$$\text{Scores} = [[0.3, 0.7, 1.1, 0.0], [0.7, 1.9, 3.1, 0.0], [1.1, 3.1, 5.1, 0.0], [0.0, 0.0, 0.0, 0.0]]$$

Look Ahead Mask

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

Padding Mask

$$\begin{bmatrix} 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Combined Mask

$$\text{Combined mask} = \text{Look Ahead} \times \text{Padding}$$

$$= [[1, 0, 0, 0], [1, 1, 0, 0], [1, 1, 1, 0], [1, 1, 1, 0]]$$

Masked Score

$$\text{Combined mask} = \begin{bmatrix} [1, -\infty, -\infty, -\infty], \\ [1, 1, -\infty, -\infty], \\ [1, 1, 1, -\infty], \\ [1, 1, 1, -\infty] \end{bmatrix}$$

Masked Score = Score . Combined Mask

$$= \begin{bmatrix} [0.3, -\infty, -\infty, -\infty], \\ [0.7, 0.9, -\infty, -\infty], \\ [1.1, 3.1, 5.1, -\infty], \\ [0.0, 0.0, 0.0, -\infty] \end{bmatrix}$$

Softmax

Softmax Score = Softmax(Masked Score)

$$= \begin{bmatrix} [1.0, 0.0, 0.0, 0.0], \\ [0.3, 0.7, 0.0, 0.0], \\ [0.1, 0.3, 0.6, 0.0], \\ [1.0, 0.0, 0.0, 0.0] \end{bmatrix}$$

Weighted Sum of Values

Attention Score = Softmax Score × Value Vector (V)

→ Encoder Decoder Multihead Attention

① From encoder output → set of attention vector K and V

② Masked multi head → Attention vector Q

These keys and values are to be used by each decoder in its "encoder-decoder" attention layer

⇓

helps the decoder to focus on appropriate places in the input sequence

→ Final Linear and Softmax Layer

① The linear layer is a simple fully connected neural network that projects the vector produced by the ~~sequence~~ stack of decoder ⇒ Logit vectors

Model ⇒ 10,000 ⇒ Vocabulary ⇒ logit vectors ⇒ 10,000

② Softmax layer turn those vector scores into probabilities. The cell / vector with the highest probability is choosen and the word associated with it is produced as the output